CIS 4360
Secure Computer Systems

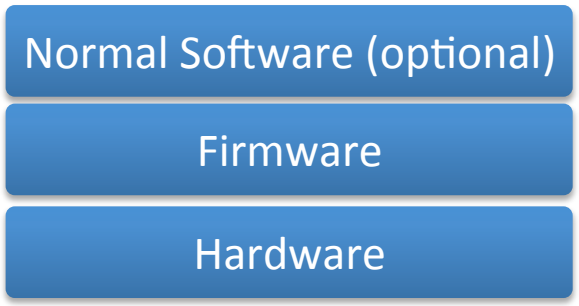
Firmware Security

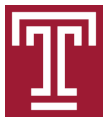
Professor Qiang Zeng

Spring 2017

Firmware

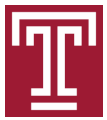
- Firmware: special software that is embedded in a hardware device and directly communicates with the device
- Almost all electronics devices run firmware
  - Examples: printers, mobile phones, routers, USB drives, medical implants, TV, cars, and traffic lights





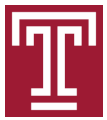
Firmware Characteristics

- Firmware is typically stored on non-volatile memory, such as EEROM (**E**lectrically **E**rasable **P**rogrammable **R**ead-**O**nly **M**emory)
- Firmware update (called *flashing*) is typically rare, and the update process is not foolproof (you may brick it)
  - E.g., DVD player companies may release new firmware to support new formats of discs. But few would got to update a DVD player
  - It means that a bug in a device's firmware may persist during the lifetime of the device

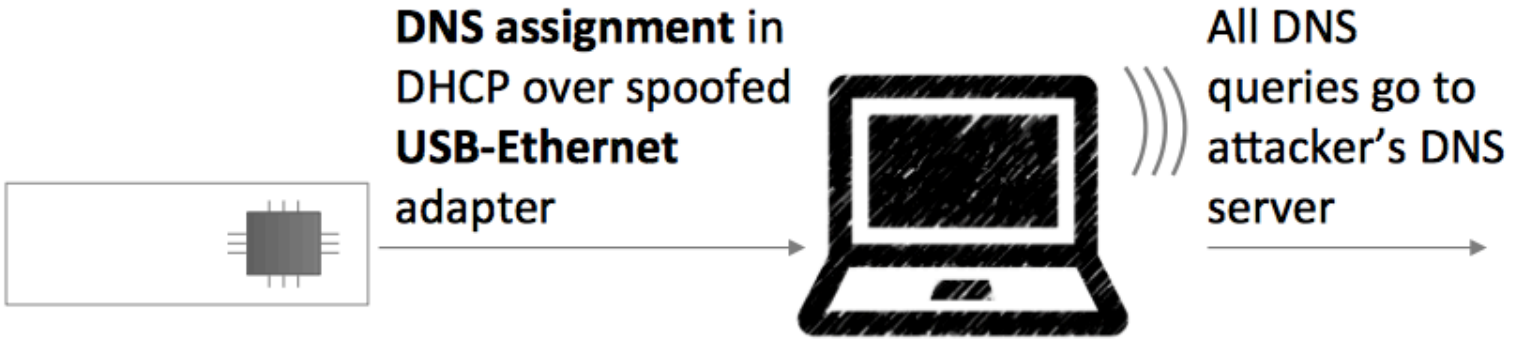


Attack measures

- Instead of exploiting a bug in firmware, however, most current hack cases modify firmware to launch attacks
- Two cases:
  - Attack firmware in USB drives
  - Attack firmware in cars



Case 1: BadUSB [Blackhat2014]



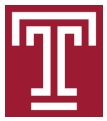
Attack steps	Result
1. USB stick spoofs Ethernet adapter 2. Replies to DHCP query with DNS server on the Internet, but without default gateway	3. Internet traffic is still routed through the normal Wi-Fi connection 4. However, DNS queries are sent to the USB-supplied server, enabling redirection attacks

No effective defenses from USB attacks exist

Protection idea	Limitation
Whitelist USB devices	▪ USB devices do not always have a unique serial number ▪ OS's don't (yet) have whitelist mechanisms
Block critical device classes, block USB completely	▪ Obvious usability impact ▪ Very basic device classes can be used for abuse; not much is left of USB when these are blocked
Scan peripheral firmware for malware	▪ The firmware of a USB device can typically only be read back with the help of that firmware (if at all): A malicious firmware can spoof a legitimate one
Use code signing for firmware updates	▪ Implementation errors may still allow installing unauthorized firmware upgrades ▪ Secure cryptography is hard to implement on small microcontrollers ▪ Billions of existing devices stay vulnerable
Disable firmware updates in hardware	▪ **Simple and effective**

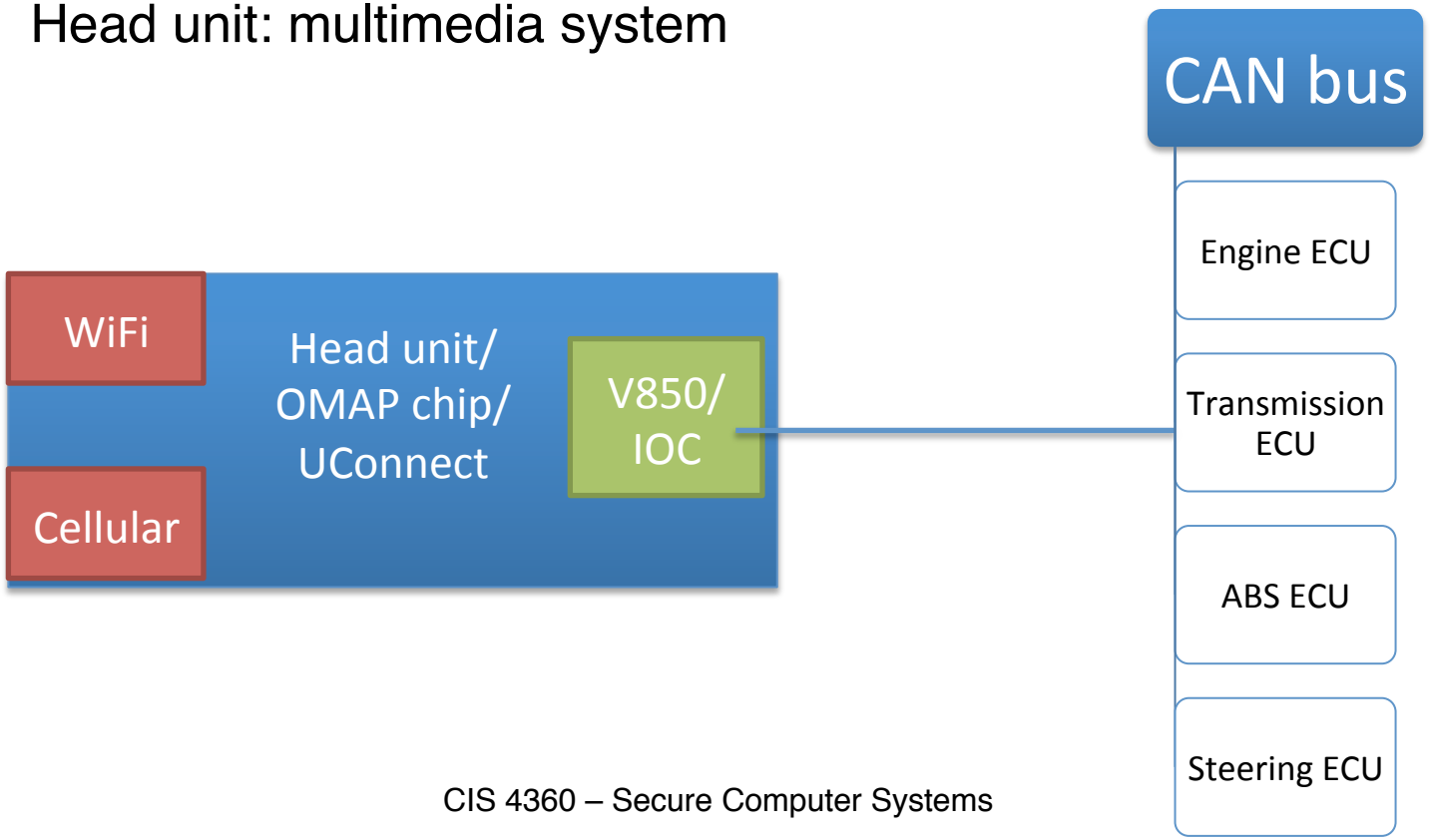
Case 2: Remote Exploitation of Cars [BlackHat'15]

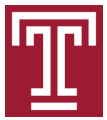
- Threat:
  - Remotely (e.g., from PA to CA) control a 2013-2015 Jeep, Ram, or Dodge
- Impact:
  - Fiat Chrysler recalled 1.4 million cars (07/2015)
  - Sprint changed its network firewall policy



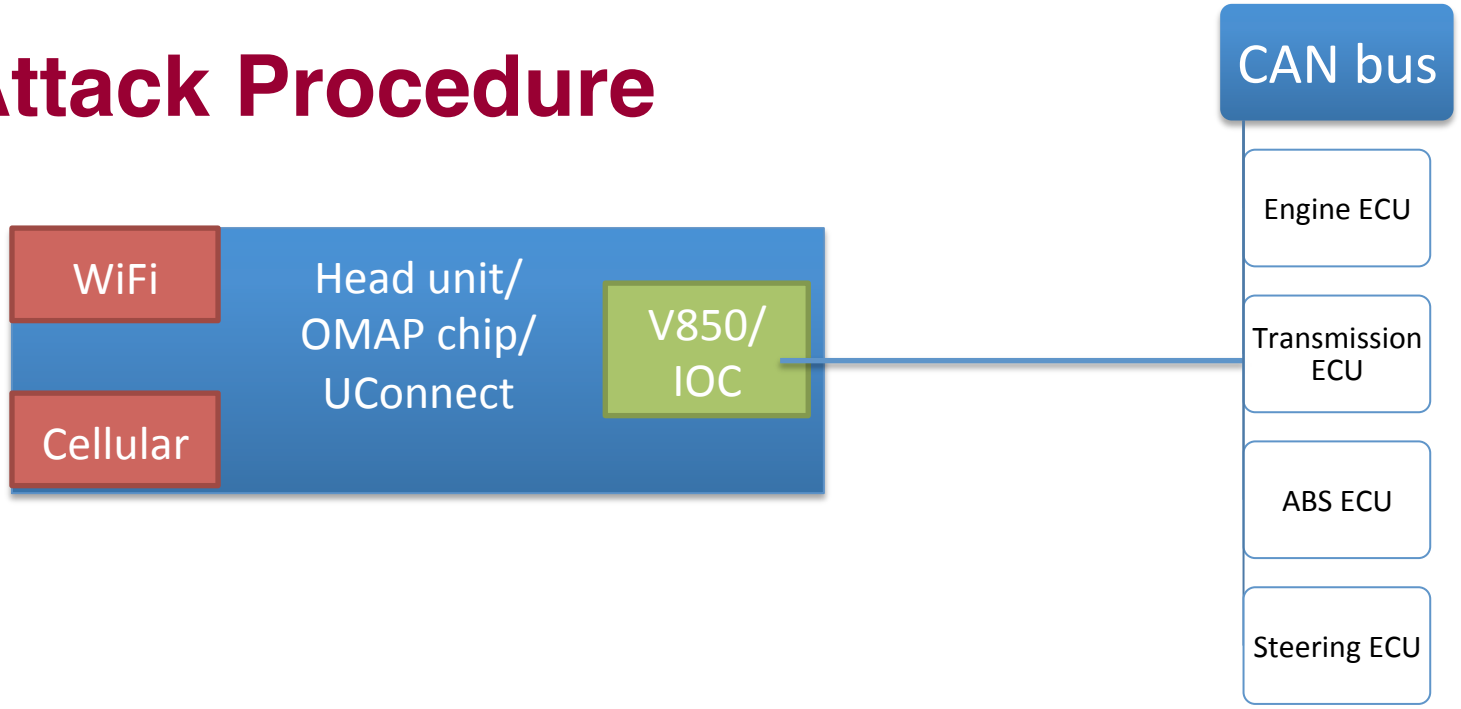
Terms and Architecture

- Terms:
  - CAN: Controller Area Network. A message bus in vehicle for inter-component communication
  - ECU: Electronic Control Unit. Each is an embedded system. E.g., engine ECU, transmission ECU, airbag ECU, ABS ECU
  - Head unit: multimedia system

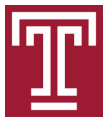




Attack Procedure

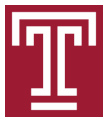


1. Establish network connection with victim car: either guess WiFi password, or scan cars connected to the Sprint cellular network
2. Port scanning and find a vulnerable service listening at some port
3. Exploit the service to login the computer for the head unit
4. Command the head unit to "update" the firmware at V850
5. Now you can send messages to the ECUs to control the car



Talk by Miller and Valasek

- https://youtu.be/OobLb1McxnI



References

- “BadUSB — On accessories that turn evil”, K Nohl, et al. BlackHat’14
- “Remote Exploitation of An Unaltered Passenger Vehicle”, C Miller and C Valasek. BlackHat’15